

Utilisation de `odeint` pour les équations différentielles du 2nd ordre

Objectif : résoudre une équation différentielle d'ordre 2 à l'aide de la fonction `odeint` de Python.

Précédemment, nous avons vu la méthode d'Euler explicite pour résoudre une équation différentielle d'ordre 1. Cette méthode n'est cependant pas toujours fiable. Nous allons ici utiliser la fonction `odeint`, de la librairie `scipy.integrate`. Elle permet de résoudre des équations différentielles de manière fiable et rapide.

De plus, nous nous intéressons à une équation **d'ordre 2**, comme celle du pendule simple :

$$\ddot{\theta} + \omega_0^2 \sin \theta = 0, \quad \text{soit aussi} \quad \boxed{\ddot{\theta} = -\omega_0^2 \sin \theta.} \quad (1)$$

a/ Reformulation de l'équation en une équation d'ordre 1

La fonction `odeint` a une syntaxe particulière, qui oblige à transformer l'équation d'ordre 2 sur θ , en une équation d'ordre 1 sur le couple $y = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix}$.

On parle parfois de vecteur à deux composantes pour désigner y :

- sa première composante est $y_0 = \theta$,
- sa seconde composante est $y_1 = \dot{\theta}$.

↪₁ Dériver le couple y pour trouver l'équation différentielle qu'il suit.

On arrive donc à une équation du premier ordre sur y :

$$\boxed{\frac{dy}{dt} = F(y) \quad \text{avec} \quad F(y) = \begin{pmatrix} y_1 \\ -\omega_0^2 \sin y_0 \end{pmatrix}.} \quad (2)$$

Bilan : on est passé de l'équation d'ordre 2 sur θ à l'équation ci-dessus, d'ordre 1 sur le couple $y = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix}$.

Autres exemples pour bien comprendre

(à faire sur feuille à part).

\rightsquigarrow_2 On considère l'équation du pendule, mais avec des frottements :

$$\ddot{\theta} = -\omega_0^2 \sin \theta - \alpha \dot{\theta}.$$

Avec la même démarche que ci-dessus, transformer cette équation d'ordre 2 sur θ en une équation d'ordre 1 sur le couple $y = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix}$ (donc en une équation du type $\frac{dy}{dt} = F(y)$).

\rightsquigarrow_3 On considère l'équation suivie par la tension u aux bornes du condensateur dans un circuit RLC en régime sinusoïdal forcé :

$$\ddot{u} + \frac{\omega_0}{Q} \dot{u} + \omega_0^2 u = \alpha \cos \omega t.$$

Avec la même démarche que ci-dessus, transformer cette équation d'ordre 2 sur u en une équation d'ordre 1 sur le couple $y = \begin{pmatrix} u \\ \dot{u} \end{pmatrix}$ (donc en une équation du type $\frac{dy}{dt} = F(y,t)$).

b/ Utilisation de odeint

On repart sur l'exemple du pendule (équations 1 et 2). on utilise la fonction `odeint` de la librairie `scipy.integrate` dont voici la syntaxe :

```
import scipy.integrate as sp
y_sol = sp.odeint(F, y_ini, t)
```

Paramètres de odeint :

- `F` est une fonction du type $F(y, t)$, qui renvoie la valeur de la dérivée de y à l'instant t . Pour résoudre une équation d'ordre 1, y est un scalaire.

Mais si l'équation est d'ordre 2, alors y est de type vecteur : $y = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$.

Attention, même si F ne dépend pas de t , il faut tout de même que t apparaisse comme second argument : $F(y, t)$.

- `y_ini` : valeur initiale de y . C'est un couple qui a autant de composantes que y .
- `t` est un tableau qui contient les instants auxquels la solution sera calculée.

Par exemple, pour le problème du pendule, on peut définir `F`, `y_ini` et `t` de la façon suivante (avant l'appel à `odeint`) :

```
# Fonction F telle que d/dt(theta, theta_point) = F(theta, theta_point)
def F(y,t):
    return (y[1], -w0**2 * np.sin(y[0]))

t = np.linspace(0,5*T0,1000) # tableau des temps, de 0 à 5T0, 1000 points.

theta0 = 50 * np.pi/180.      # angle initial (rad), ici correspond à 50 degrés.
theta_prime0 = 0               # vitesse angulaire initiale (rad/s)
y_ini = (theta0, theta_prime0) # valeur initiale de y = (theta, theta_point)
```

C'est seulement une fois `F`, `y_ini` et `t` définis comme ci-dessus qu'on peut exécuter `odeint` :

```
y_sol = sp.odeint(F, y_ini, t)
```

Objets retournés par odeint :

- Après exécution de `y_sol = sp.odeint(F, y_ini, t)`, `y_sol` est une matrice dont la première colonne contient les valeurs de $y_0(t)$ (donc dans notre exemple, de $\theta(t)$), et la seconde colonne contient les valeurs de $y_1(t)$ (donc dans notre exemple, de $\dot{\theta}(t)$).

On peut y accéder avec la syntaxe suivante :

```
theta = y_sol[:,0]      # 1ère colonne de y_sol
theta_prime = y_sol[:,1] # 2ème colonne de y_sol
```

Il suffit ensuite de tracer `theta` en fonction de `t` avec les méthodes usuelles :

```
plt.plot(t,theta, label="solution de l'équation non linéaire avec odeint"),
etc.
```

c/ Exemple d'utilisation dans le cas du pendule

1 - Reprendre tous les éléments précédents pour écrire un algorithme qui permet de résoudre l'équation non linéaire du pendule (équation 1, que l'on a transformée en l'équation 2). On tracera la solution.

Des éléments de code sont déjà présents dans le fichier Capytale numéro dac9-1273229.

2 - Écrire une fonction `solution_petits_angles(t, theta0)` qui prend en argument un instant `t` et un angle initial `theta0`, et qui retourne la valeur `theta0 * np.cos(w0*t)` (où `w0` est la pulsation, déjà définie par ailleurs).

3 - Ajouter, sur le même graphe que celui de la question 1, la solution pour les petits angles. Faire pour cela une nouvelle ligne avec `plt.plot`.

4 - Explorer avec différents angles initiaux. Tester par exemple 5° , 45° et 140° .

À partir de quel angle voit-on clairement un désaccord entre la solution sans approximation (équation avec $\sin \theta$) et la solution avec petits angles (approximation $\sin \theta \simeq \theta$) ?

5 - À l'aide de votre simulation numérique, mesurer la période T des oscillations pour des angles initiaux de 10° , 30° , 60° et 80° . Calculer le rapport T/T_0 dans chaque cas (où $T_0 = 2\pi/\omega_0$ est la période pour les petits angles) et consigner les résultats dans un tableau sur votre feuille.

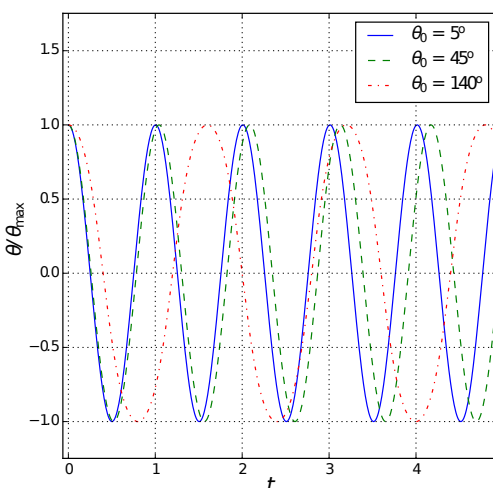
6 - On souhaite vérifier ceci **expérimentalement**, à l'aide du pendule.

Faire d'abord la mesure de T_0 en prenant un angle petit (10° par exemple).

Faire ensuite des mesures à 30° , 60° et 80° , calculer le rapport T/T_0 , et comparer avec les résultats numériques : y a-t-il accord ?

Vous pouvez essayer pour des angles plus grands, mais vous verrez que le capteur pose problème et qu'il faut ruser.

7 - Si le temps le permet, recommencer la mesure de la période T pour d'autres angles initiaux θ_0 , afin de tracer une courbe qui donne T en fonction de θ_0 (faire une dizaine de points). Comparer à la formule théorique de Borda (qui est une approximation) : $T = T_0 \left(1 + \frac{\theta_0^2}{16} + \frac{11}{3072} \theta_0^4 \right)$ en la traçant sur le même graphique.



Exemple de solutions numériques obtenues avec odeint, sans approximation, pour trois conditions initiales différentes. Cf question 1 ci-dessus.